mgr inż. Jerzy MOCAŁA
Instytut Maszyn Matematycznych

System BASIC MERA 400 i WANG 2200. Porównanie

Wstęp

Rozwój produkcji zestawu MERA-400 i wprowadzenie go do biur projektowych spowodował rozpoczęcie prac nad przenoszeniem oprogramowania z innych maszyn. Bogate oprogramowanie maszyny cyfrowej WANG 2200 w BASIC-u oraz łatwość korzystania z programów na tej maszynie przez użytkowników-projektantów spowodowała, że użytkownicy mo MERA-400 chcą uzyskać podobne wygodne narzędzie do prac projektowych. Przed rozpoczęciem prac implementacyjnych programów z mo WANG na mo MERA warto zdać sobie sprawę z różnic tych systemów, aby założenie, że wystarczy zmienić kod programów nie okazało się błędne. Różnice mogą występować nie tylko w formie pewnych instrukcji, ale także w reprezentacji maszynowej liczb i wyrażeń, lub w interpretacji instrukcji.

W artykule przedstawiono podstawowe różnice systemów BASIC dla minikomputerów MERA-400 i WANG 2200. Przyjmując BASIC na WANG za system wzorcowy - przedstawiamy encyklopedyczny opis systemu BASIC dla mo MERA-400.

Ponadto opisując dyrektywy i instrukcje języka przyjęto następujące zasady:

- parametry w nawiasach trójkątnych < > muszą być zastąpione wartościami podanymi przez użytkownika zgodnie z jego intencją wykonania prac przez mo,
- nawiasy kwadratowe [ ] oznaczają, że parametry w nich zawarte są opcjonalne,
- parametry zawarte w klamrach { } umieszczone jeden nad drugim oznaczają, że jeden z nich jest wymagany,
- nawiasy z wielokropkiem oznaczają, że parametr w nich zawarty może powtarzać się wielokrotnie,
- prostokąty oznaczają klawisze zwrotnego urządzenia wejścia/wyjścia np. CR oznacza klawisz "powrót karetki".

Ogólny opis systemu BASIC

System BASIC jest systemem konwersacyjnym umożliwiającym użytkownikowi pracę w trybie natychmiastowym i w trybie programowym.

W trybie natychmiastowym wykonuje się dyrektywy typu rezerwacji obszaru na dysku, wyświetlania wartości zmiennych programu, których obliczeń itp. W trybie programowym wykonuje się przygotowany program w języku programowania BASIC.

Te dwa tryby wzajemnie się przeplatają. Należy przy tym zwrócić uwagę, że część dyrektyw ma swoje odpowiedniki w postaci instrukcji języka BASIC.

Program w języku BASIC zbudowany jest z wierszy składających się z numeru wiersza i z instrukcji.

Numer wiersza określa kolejność wykonywania instrukcji. Instrukcje języka BASIC są analogiczne do innych języków algorytmicznych, np. do FORTRAN-u.

Program w języku BASIC wprowadza się do pamięci maszyny wiersz po wierszu, przy czym kolejność wprowadzania wierszy jest dowolna, nie musi się pokrywać z kolejnością wynikającą z porządku ustalonego przez wzrastające numery wierszy. Jeśli wiersz programu jest syntaktycznie poprawny, zostanie zapamiętany, stanowiąc część tworzonego programu.

Poprawność całego programu sprawdza się podczas realizacji dyrektywy systemowej wykonania programu.

Konwersacyjność(użytkownik komunikuje się z systemem przez zwrotne urządzenie wejścia/wyjścia) umożliwia przerwanie wykonania programu, zlecanie wykonania dyrektyw i następnie powracanie do zawieszonego programu i kontynuowanie jego pracy od momentu przerwania. Ułatwia to uruchamianie programu, a także pozwala uzyskać wygodną formę użytkową programu.

Dyrektywy systemu BASIC. Tryb natychmiastowy

Dyrektywy są to zlecenia użytkownika wykonywane w trybie natychmiastowym. System przechodzi do pracy w trybie natychmiastowym, jeśli wprowadzony wiersz informacji nie ma numeru.

Zakres prac możliwych do realizacji za pomocą dyrektyw jest następujący:

- listowanie programu przechowywanego w pamięci operacyjnej,
- oczyszczenie pamięci operacyjnej,
- załadowanie programu do pamięci operacyjnej z pamięci zewnętrznej (dysk, taśma),
- przesłanie programu z pamięci operacyjnej do pamięci zewnętrznej,
- przenumerowanie programu,
- zatrzymanie i kontynuacja programu,
- nadanie wierszowi kolejnego numeru.

Formalny zapis tych dyrektyw dla systemów WANG i MERA przedstawia tab. 1.

Tab. 1. Podstawowe dyrektywy systemu BASIC

Opis ogólny	WANG 2200	MERA 400
Listowanie programu	LIST [S] [⟨nr1⟩ [, ⟨nr2⟩]]	LIST [# ⟨n⟩ [⟨nr1⟩ - - [⟨nr2⟩]]]
Czyszczenie pamięci operacyjnej	CLEAR [{P / V / N}] [⟨nr1⟩ [, ⟨nr2⟩]]	{NEW / SCRATCH}
Załadowanie programu do PAO z pamięci zewnętrznej	LOAD [DC {F / R / T}] [{#n / /XXX}] ["nazwa"] [⟨nr1⟩] [, ⟨nr2⟩]]	{LOAD #⟨n⟩ / OLD ⟨z⟩}
Przesłanie programu z PAO do pamięci zewnętrznej	SAVE [DC {F / R / T}] [S] [(wyrażenie / "nazwa")] [{#n / /XXX}] [P] ["nazwa"] [⟨nr1⟩] [, ⟨nr2⟩]]	SAVE
Przenumerowanie programu	RENUMBER [⟨nr1⟩] [, ⟨nr2⟩] [, ⟨LICZBA⟩]	RESEQUENCE [⟨nr1⟩] [, ⟨nr2⟩] [, ⟨liczba⟩]

Opis ogólny	WANG 2200	MERA 400
Zatrzymanie programu	{ HALT STEP / RESET }	CONTROL i T
Wznowienie programu	{ CONTINUE / GOTO <nr> / RUN [<nr>] }	RUN [# nr1]. [,# nr2] [, <nr1> - <nr2>]
Nadanie wierszowi kolejnego numeru	STMT NUMBER	
Czyszczenie dysku	SCRATCH {F/R/T} [{#n,/XXX,}] "<nazwa1>" [,"<nazwa2>"] ...	-
Listowanie dysku	LIST DC {F/R/T} [{#n/XXX}]	-
Przepisywanie zawartości talerzy dyskowych	MOVE [{#n,/XXX,}] { RF / FR }	-
Sprawdzanie poprawności zapisu na dysku	VERIFY {F/R/T} [{#n,/XXX,}] [(<wyrażenie1>,<wyrażenie2>)]	-
Rezerwacja obszaru na dysku	DATA SAVE DC OPEN {F/R/T} [$] [#n] { {"<nazwa>" / "<wyrażenie>"}, '<nazwa> / TEMP, '<wyrażenie1>' , '<wyrażenie2>' }	-

#n - numer logiczny, do którego jest przypisany adres urządzenia zewnętrznego,
/XXX - adres w kodzie szesnastkowym urządzenia zewnętrznego
R - talerz wymienny,
F - talerz stały

W podanym zestawieniu podano jedynie podstawowe dyrektywy. Ilustruje ono różnice formalne. Dokładniejszy opis poszczególnych dyrektyw przedstawi różnice w działaniu i sytuacje, w których korzysta się z tych dyrektyw.

- Listowanie programu

Dyrektywa LIST służy do listowania programu znajdującego się w pamięci operacyjnej. Urządzenie, na którym będzie wyświetlany drukowany listing programu, zależy od dyrektywy:

SELECT LIST <nr> [(<długość>)]

gdzie: nr - numer urządzenia,
długość - maksymalna długość wiersza na wydruku.
Dyrektywa ta wybiera odpowiednie urządzenie wyjściowe.

Przykład

SELECT LIST 215 (116) - specyfikuje drukarkę z maksymalną długością wiersza na wydruku 116 znaków,

SELECT LIST ØØ5 (5Ø) - specyfikuje monitor ekranowy z maksymalną długością wyświetlanego wiersza 50 znaków (ekran ma standardową długość 64 znaki).

Parametr S w dyrektywie LIST przeznaczony jest do użycia przy wyświetlaniu programu na monitorze ekranowym. Powoduje on listowanie programu stronami. Strona zawiera tyle wierszy programu, ile mieści się na ekranie. Po każdej stronie wyświetlanie się kończy. W celu kontynuacji listingu trzeba nacisnąć klawisz CR/LF, co spowoduje wyświetlenie następnej strony.

Parametry nr 1, nr 2 oznaczają część programu zawartą między tymi numerami, która ma być listowana.

Dyrektywa MERA-400 działa podobnie. Parametr n określa urządzenie, na którym chcemy otrzymać listing. Brak w niej parametru S, co w sytuacji, gdy system korzysta z monitorów ekranowych jako urządzeń WE/WY powoduje, że zawsze listing programu należy wyprowadzać na drukarkę. W przeciwnym wypadku dostępne będą na ekranie jedynie ostatnie wiersze programu. Może to być utrudnieniem, gdy monitory są zainstalowane w innym pomieszczeniu niż drukarka.

Czyszczenie pamięci operacyjnej

Dyrektywa CLEAR na WANG-u ma różnorodne zastosowanie w zależności od użytych parametrów:

CLEAR - usuwa z pamięci program wraz ze zmiennymi oraz zwalnia wszystkie przypisane urządzenia WE/WY,

CLEAR P - usuwa tylko tekst programu,

CLEAR V - zeruje wszystkie zmienne występujące w programie,

CLEAR N - zeruje zmienne, które nie występują w obszarze COMMON.

Parametry nr 1 i nr 2 określają granice działania dyrektywy.

Podanie numerów wierszy ogranicza działanie dyrektywy do części programu zawartej między tymi wierszami.

Dyrektywy MERA-400 działają jak dyrektywa CLEAR bez parametrów w systemie BASIC WANG:

NEW ≡ SCRATCH ≡ CLEAR

Ładowanie programu do pamięci operacyjnej

Dyrektywa LOAD umożliwia wprowadzenie programu przechowywanego w pamięci zewnętrznej do pamięci operacyjnej. Jeżeli w PAO znajduje się program, to tworzona jest suma programów tego, który był w pamięci i nowo ściąganego.

Przykład

LOAD DC R "CDPRO" 1ØØ,2ØØ - w miejsce od numeru wiersza 100 do 200 zostanie umieszczony program przechowywany na dysku wymiennym pod nazwą CDPRO (por. instrukcja LOAD),

LOAD "PROG1" - z pamięci taśmowej (kasetowej) zostanie ściągnięty program przechowywany pod nazwą PROG1.

W systemie BASIC MERA-400 dyrektywa ładowania jest zróżnicowana w zależności od nośnika, na którym przechowywany jest program.

Dyrektywa ładowania programu z dysku ma postać:

OLD < z >

Oznacza ona ładowanie do wyczyszczonej pamięci operacyjnej programu ze zbioru dyskowego o numerze z.

Dyrektywa

LOAD # < n >

o parametrze n - numer logiczny urządzenia wejściowego (dysk, czytnik) ściąga do pamięci program bez czyszczenia pamięci. Działa więc podobnie jak dyrektywa WANG-owska. Ponieważ podaje się w niej jedynie numer logiczny urządzenia WE potrzeba wcześniej "ustawić" na nim potrzebny program.

Przesłanie programu z pamięci operacyjnej

Przesłanie całego programu lub jego części do pamięci zewnętrznej można zrealizować za pomocą dyrektywy SAVE.

Przykład

SAVE DC F ("STARY") "NOWY" - oznacza przesłanie programu NOWY z pamięci na dysk stały na miejsce programu STARY,

SAVE "PGM" 1ØØ,5ØØ - oznacza przesłanie części programu PGM (od wiersza 1ØØ do 5ØØ) na taśmę magnetyczną.

W systemie BASIC MERA-400 SAVE jest dyrektywą bezparametryczną. Jej zastosowanie ogranicza się jedynie do możliwości przepisania programu z pamięci na zbiór dyskowy. Program dopisywany jest na końcu pliku zbiorów. Aby móc się do niego odwołać należy zapamiętać miejsce zapisu (zob. dyrektywę OLD).

Przenumerowanie programu

System BASIC umożliwia aktualizację programu np. przez wstawienie nowych wierszy. Może się zdarzyć, że sąsiednie wiersze mają numery nie pozwalające wstawić dodatkowego ciągu wierszy. Niezbędne jest przenumerowanie programu nadanie nowych numerów wierszom oraz aktualizacja skoków itp. . Do tego celu służy dyrektywa RENUMBER.

Parametr nr 1 oznacza numer wiersza, od którego ma być prowadzona renumeracja, nr 2 oznacza numer, na jaki ma być zamieniony numer nr 1, liczba określa krok numeracji.

Dyrektywa RESEQUENCE spełnia podobne funkcje. Różnice występują przy pomijaniu parametrów dyrektywy.

Dla WANG-a dyrektywa postaci:
RENUMBER - spowoduje nadanie całemu programowi numerów co 10 zaczynając od 10.

Natomiast dla MERA-400 dyrektywa:
RESEQUENCE - spowoduje nadanie całemu programowi numerów co 10 zaczynając od 100.

Zatrzymanie programu

W systemie BASIC istnieje możliwość zatrzymania wykonywanego programu przez naciśnięcie klawisza [HALT/STEP] lub [RESET].

Klawisz [HALT/STEP] przeważnie używany jest do wykonywania programu krok po kroku. Odnosi się on do jednej instrukcji programu. Jeśli w jednym wierszu programu było kilka instrukcji, to program zatrzymuje się po wykonaniu przetwarzanej instrukcji. Dyrektywa [RESET] jest używana, gdy w wypadku zakłócenia przerwanie wykonania programu przez [HALT/STEP] nie skutkuje.

Dyrektywa ta nie pozwala na kontynuację wykonywania programu od miejsca, w którym nastąpiło przerwanie, chociaż tekst programu z aktualnie obliczonymi wartościami zmiennych przechowywany jest w pamięci operacyjnej.

W systemie BASIC MERA zatrzymanie działania programu jest możliwe przez wciśnięcie klawiszy [CONTROL] i [T]. Ponieważ nie można wznowić programu od miejsca przerwania, dyrektywa ta jest odpowiednikiem dyrektywy [RESERT].

Wznowienie działania programu

Rozpoczęcie wykonywania programu dokonuje się za pomocą dyrektywy RUN. Jeśli nie użyto parametru, program wystartuje od wiersza o najniższym numerze. Przed przekazaniem sterowania programowi, zerowane są zmienne spoza obszarów COMMON (por. Instrukcje deklaracji). Dyrektywa tej postaci służy do rozpoczęcia wykonania programu załadowanego do pamięci. Gdy przerwano działanie programu (np. dyrektywą HALT/STEP), a następnie program ma kontynuować obliczenia od miejsca przerwania, należy użyć dyrektyw:

CONTINUE , GOTO < nr >, RUN < nr > .

Programu nie można kontynuować, gdy po przerwaniu dołączono nowy wiersz, przenumerowano program, wykonano dyrektywę CLEAR. Dyrektywa CONTINUE powoduje wznowienie wykonywanego programu od następnej instrukcji, przy której nastąpiło przerwanie.

Dyrektywy GOTO < nr >, RUN < nr > wznawiają wykonanie od numeru wiersza podanego jako parametr dyrektywy. Nie może to być numer wiersza występujący w pętli FORT-NEXT lub podprogramie GOSUB-RETURN.

System BASIC MERA nie ma tak bogatego aparatu wznawiającego działanie programu.

Dyrektywa RUN zeruje zawsze zmienne proste, więc jedynie może służyć do wystartowania całego lub części programu.

Znaczenie parametrów dyrektywy jest następujące:

n1 - numer logiczny wejścia,
n2 - numer logiczny wyjścia,
nr 1 - nr 2 - numery wierszy, które ograniczają wykonanie programu od nr 1 do nr 2.

Kolejny numer wiersza

Numery wierszy mogą być wprowadzane bądź z klawiatury numerycznej, bądź za pomocą klawisza STMT NUMBER .

Naciśnięcie klawisza powoduje nadanie wierszowi, który ma być wprowadzany, numeru większego o 10 od numeru ostatnio wprowadzonego wiersza.

W systemie BASIC MERA numery wierszy wprowadza się tylko z klawiatury numerycznej.

Operacje dyskowe

Ze względu na to, że w obecnej postaci system BASIC MERA-400 nie ma możliwości korzystania z dysku, nie wydaje się celowe omawianie szerzej dyrektyw związanych z pamięcią dyskową.

W systemie BASIC oprócz dyrektyw, można w trybie natychmiastowym wykonywać pewne prace (nazywane dalej zleceniami) typu: wydrukować wartość zmiennych, nadać nowe wartości zmiennym, obliczyć proste wyrażenie arytmetyczne, itp.

W systemie MERA-400 jest możliwe do wykonania jedynie zlecenie drukowania wartości zmiennych lub wyrażenia arytmetycznego, jeśli nie będą to zmienne indeksowane lub funkcje definiowalne.

Postać zlecenia jest następująca:

PRINT { < wyrażenie arytmetyczne > / < zmienna prosta > }

Przykład

PRINT A*B - na urządzeniu zwrotnym WE/WY wydrukowana zostanie wartość iloczynu, jeśli poprzednio zmienne A i B występowały w programie,

PRINT A (3) - zlecenie niepoprawne, zawiera element tablicy.

Dla porównania zlecenie tej klasy na WANG-u może mieć postać:

<instrukcja> : <instrukcja> : ..... : PRINT <wyrażenie arytmetyczne>

Pozwala to wykonywać proste programy mieszczące się w jednym wierszu. Praca na maszynie cyfrowej podobna jest wtedy do obliczeń wykonywanych na kalkulatorze elektronicznym.

Opis języka BASIC. Tryb programowy

W systemie BASIC w trybie programowym wykonuje się programy napisane w języku BASIC. Elementami składowymi programu w języku BASIC są instrukcje, stałe, zmienne proste i tablice, wyrażenia arytmetyczne, numery wierszy, funkcje standardowe, operatory arytmetyczne i logiczne oraz spacje.

Poszczególne te elementy zostaną omówione poniżej.

Podstawowe symbole języka

Litery: od A do Z (26 znaków), w systemie MERA do liter dołączono znak @ (27 znaków).
Cyfry: od Ø do 9.
Operatory arytmetyczne: + - * / ↑
Operatory relacji: = > < => =< < >
Ograniczniki: , ; : " ' (w systemie MERA nie ma ogranicznika ')
Kropka dziesiętna: .
Nawiasy: ( )
Znak zapytania: ?
Znak numeru: #
Znaki specjalne: $ % (w systemie MERA znaki te można używać w komentarzach i łańcuchach alfanumerycznych).

Typy stałych

W programach w języku BASIC można używać stałych liczbowych i alfanumerycznych. Stałe liczbowe mogą być przedstawione w postaci liczb całkowitych, ułamków dziesiętnych lub w postaci z wykładnikiem, ale zawsze są traktowane przez maszynę jako liczby zmiennoprzecinkowe. Stała liczbowa może mieć co najwyżej 13 cyfr (w systemie MERA - 11 cyfr). Dokładność i zakres liczb reprezentowanych przez maszynę cyfrową WANG wynosi:

$$1E-99 < <\text{liczba}> < 1E+99$$

W systemie MERA istnieją dwie funkcje standardowe, które informują o dokładności i zakresie liczb.

Funkcja EPS jest najmniejszą liczbą, która spełnia w arytmetyce maszyny nierówność:

$$1+EPS > 1 \quad \text{i} \quad 1-EPS < 1$$

EPS=1.81899 E-12

Funkcja INF jest największą liczbą, dla której +INT oraz -INF da się przedstawić w maszynie cyfrowej:

INF=1.70141 E+38

Dla wygody programistów w BASIC-u wprowadzono standardowo stałą liczbową oznaczoną # PI, której wartość, to liczba π - na mo MERA stałą tę oznacza się następująco: @ PI.

Oprócz stałych liczbowych można używać stałych alfanumerycznych - są to łańcuchy znaków alfanumerycznych ograniczonych cudzysłowem np.: "STAŁE ALFANUMERYCZNE".

Zmienne

Zmienna jest nazwą, mającą w systemie wartość, która w trakcie wykonywania programu może się zmieniać. Ze względu na wartość rozróżnia się zmienne liczbowe i zmienne alfanumeryczne. Wartościami zmiennych liczbowych są liczby, wartościami zmiennych alfanumerycznych są łańcuchy znaków alfanumerycznych.

Ze względu na strukturę rozróżnia się zmienne proste i tablice. Możliwe rodzaje zmiennych w systemie BASIC przedstawiają tab. 2 i 3 (+ oznacza występowanie zmiennej rodzaju określonego przez podział ze względu na wartość i strukturę).

Tab. 2 Zmienne w systemie WANG

Typ zmiennej	liczbowa	alfanumeryczna
prosta	+	+
tablica	+	+

Tab. 3 Zmienne w systemie MERA

Typ zmiennej	liczbowa	alfanumeryczna
prosta	+	-
tablica	+	-

Z przedstawionego zestawienia wynika, że system BASIC MERA-400 nie ma zmiennych o wartościach alfanumerycznych.

Identyfikator (nazwa) zmiennej prostej liczbowej ma postać:

$$\langle \text{litera} \rangle \left[\langle \text{cyfra} \rangle \right]$$

Program może zatem rozróżnić 286 różnych zmiennych w systemie WANG i 297 zmiennych w systemie MERA.

Identyfikator tablicy liczbowej ma postać:

$\langle \text{litera} \rangle \left[\langle \text{cyfra} \rangle \right]$ - w systemie WANG

$\langle \text{litera} \rangle$ - w systemie MERA

Zatem różnych nazw tablic może być 286 (WANG) i 27 (MERA).

W tab. 4 przedstawiono możliwe identyfikatory zmiennych.

Tab. 4. Identyfikatory zmiennych w języku BASIC

Rodzaj zmiennej	WANG	MERA
zmienna prosta liczbowa	$\langle \text{litera} \rangle \left[\langle \text{cyfra} \rangle \right]$	$\langle \text{litera} \rangle \left[\langle \text{cyfra} \rangle \right]$
zmienna prosta alfanumeryczna	litera $\left[\langle \text{cyfra} \rangle \right] \$$	-
tablica liczbowa	litera $\left[\langle \text{cyfra} \rangle \right]$	$\langle \text{litera} \rangle$
tablica alfanumeryczna	litera $\left[\langle \text{cyfra} \rangle \right] \$$	-

Tablice mogą być co najwyżej dwuwymiarowe. W systemie WANG indeksy tablic mogą mieć wartość od 1 do 255, natomiast na mc MERA od 0 do 32767.

Wyrażenia arytmetyczne

Wyrażenie arytmetyczne zbudowane jest ze zmiennych, stałych, operatorów arytmetycznych, funkcji standardowych, funkcji użytkownika oraz nawiasów ustalających porządek wykonywania działań.

<wyrażenie arytmetyczne> ::= <zmienna liczbowa>|<stała liczbowa>|<funkcja>|<kombinacja tych elementów połączonych znakami działań arytmetycznych>

$$\text{<zmienna liczbowa>} ::= \left\{ \begin{array}{l} \text{<zmienna liczbowa prosta>} \\ \text{<tablica liczbowa>} \end{array} \right\}$$

Kolejność wykonywania działań (gdy nie ma nawiasów) jest następująca:

1) potęgowanie
2) mnożenie i dzielenie
3) dodawanie i odejmowanie

Dla operatorów z tej samej grupy priorytetowej kolejność wykonania ustala pierwszeństwo występowania w wyrażaniu "czytając" go od lewej do prawej.

Przykład

Wyrażenie postaci: A ↑ B/C * D
jest równoważne: ((A ↑ B) /C) * D

Oznaczenie operatorów arytmetycznych przedstawiono w tab. 5.

Tab. 5. Operatory arytmetyczne

Znaczenie	Symbol operatora WANG	Symbol operatora MERA
dodawanie	+	+
odejmowanie	-	-
mnożenie	*	*
dzielenie	/	/
potęgowanie	↑	^

Instrukcja podstawienia

Instrukcja podstawienia LET pozwala zmieniać wartości poszczególnym zmiennym programu. Postać ogólna tej instrukcji w systemie WANG jest następująca:

$$\left[\text{LET}\right] \text{<zmienna>} \left[\left\{ ,\text{<zmienna>} \right\} \ldots \right] = \left\{ \begin{array}{l} \text{<wyrażenie arytmetyczne>} \\ \text{<zmienna>} \\ \text{<stała>} \end{array} \right\}$$

Przykład

100 LET A,B,C=100*A/D

Instrukcja ta oznacza, że zmiennym prostym A,B,C nadano wartość wyrażenia 100*A/D.

W systemie MERA instrukcja podstawienia odnosi się tylko do zmiennych liczbowych. Postać ogólna w tym systemie jest następująca:

$$\left[\text{LET}\right] \langle \text{zmienna liczbowa} \rangle \left[\left\{ = \langle \text{zmienna liczbowa} \rangle \right\} \ldots \right] = \left\{ \begin{array}{l} \langle \text{wyrażenie arytmetyczne} \rangle \\ \langle \text{zmienna liczbowa} \rangle \\ \langle \text{stała liczbowa} \rangle \end{array} \right\}$$

Przykład

Instrukcja z przykładu poprzedniego w systemie MERA ma postać:

```
100 LET A=B=C=100 * A/D.
```

Słowo LET może być pominięte dla oszczędności miejsca w pamięci i z reguły podczas pisania programu bywa pomijane.

Instrukcje skoku

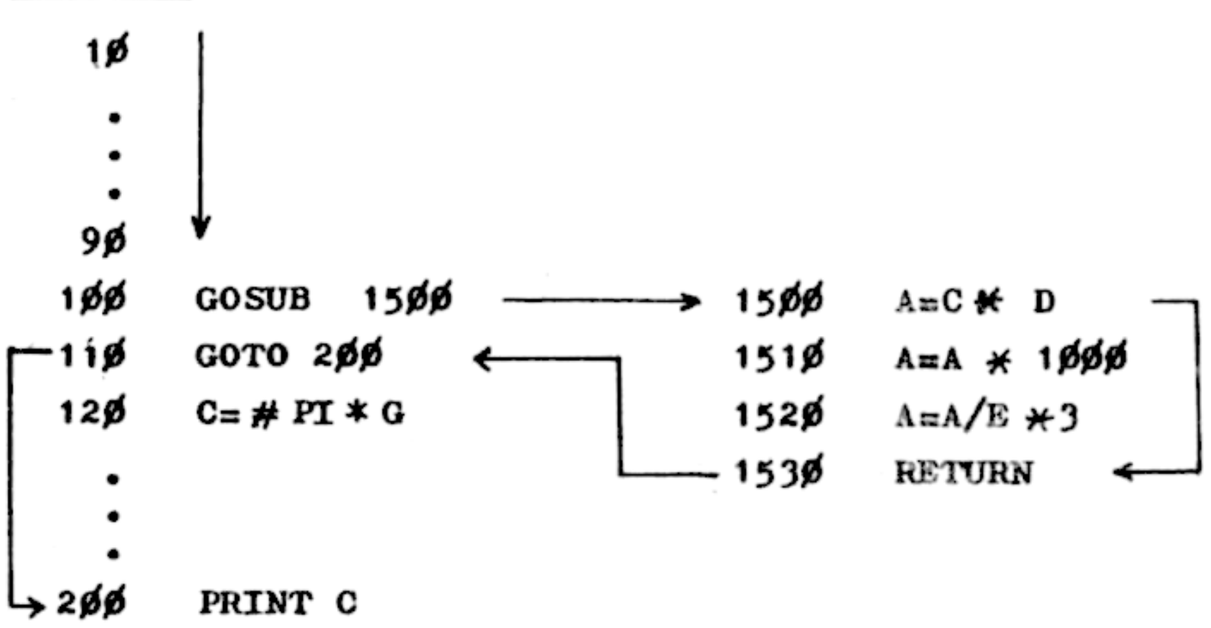
Instrukcje skoku pozwalają zmieniać kolejność wykonywania wierszy programu w czasie jego realizacji. Do tej klasy instrukcji należą:

- instrukcja skoku GOTO < nr wiersza>
- instrukcja skoku ze śladem GOSUB <nr wiersza>
- instrukcja skoku liczonego ON <wyrażenie arytmetyczne> $\left\{ \begin{array}{l} \text{GOSUB} \\ \text{GOTO} \end{array} \right\}$ <nr1> $\left[\left\{ ,\langle \text{nr2} \rangle \right\} \ldots \right]$

Instrukcja skoku służy do zmiany normalnej kolejności wykonywania programu. Określa, że następnym wykonywanym wierszem ma być wiersz podany w instrukcji skoku.

Instrukcja skoku ze śladem umożliwia wykonanie skoku do wiersza o podanym numerze, wykonanie instrukcji począwszy od tego wiersza, aż do napotkania instrukcji RETURN i powrót do wykonywania następnej instrukcji po GOSUB. Sytuację taką obrazuje poniższy przykład.

Przykład



Program wykonuje się w formie realizacji wierszy kolejno od 10 do 90, instrukcja GOSUB zmienia kolejność wykonywania. Następuje skok do wiersza 1500, teraz wykonują się wiersza od 1500 do 1530, aż do instrukcji RETURN, która powoduje powrót do wykonywania instrukcji wiersza o numerze 110. Ponieważ w tym wierszu jest instrukcja skoku, następnym realizowanym wierszem jest wiersz o numerze 200.

Instrukcja skoku liczonego jest odmienną instrukcji GOSUB lub GOTO. Podczas wykonywania instrukcji oblicza się wartość wyrażenia arytmetycznego. Część całkowita wskazuje na liście numerów wierszy pozycję, do której przekazane zostanie sterowanie.

Przykład

```
1ø   A=2
2ø   B=1.2
3ø   ON A * B   GOTO   2øø,3øø,4øø
```

Wartość wyrażenia A * B=2.4. Część całkowita wynosi 2. Na liście wierszy druga pozycja, to numer wiersza 3øø. Sterowanie zostaje przekazane do wiersza o numerze 3øø.

Jeśli ozęść oałkowita wyrażenia jest wiąksza lub mniejsza od liozby wyspeoyfikowanyoh numerów wierszy w instrukoji ON, to wykonywana jest następna instrukoja po ON. W systemie MERA sytuaoja taka traktowana jest jako błąd i przerywane jest wykonywanie programu z wydrukiem komunikatu o błędzie.

Instrukoja pętli

Instrukoja o postaoi:

FOR <liozbowa zmienna prosta> = <wyrażenie1> TO <wyrażenie2> [STEP <wyrażenie 3>]

wspólnie z instrukoją:

NEXT <liozbowa zmienna prosta>

służy do oyklioznego wykonywania instrukoji zawartyoh między tymi instrukojami.

Instrukoje będą się wykonywały tyle razy, aż <liozbowa zmienna prosta>, nazywana zmienną sterującą, nie przekroozy wartośoi określonej przez <wyrażenie2>. Za każdym przebiegiem pętli zmienna sterująoa zwiększa swoją wartość o wartośoi kroku ustaloną przez <wyrażenie3>. Gdy opoja STEP <wyrażenie3> jest pominięta, przyjmuje się wartość kroku za +1.

W systemie WANG pętla wykona się przynajmniej jeden raz, nawet w sytuaoji poniższej:

```
FOR I=1 TO-1
  :
NEXT I
```

Wartość zmiennej sterująoej przekroozyła ograniozenie, ale pętla wykona się jeden raz zanim zostanie sprawdzony warunek przerywająoy wykonanie pętli. W systemie MERA w takiej sytuaoji pętla się nie wykona, będzie pominięta. Wykona się wiersz następny za NEXT I.

Instrukoja warunkowa

Instrukoja warunkowa zmienia kolejność wykonywania wierszy programu w zależnośoi od spełnienia relaoji.

Postać ogólna instrukoji jest następująca:

IF <operand> <operator relaoji> <operand> THEN <nr wiersza>

$$\text{<operand>} ::= \left\{ \begin{array}{l} \text{<zmienna>} \\ \text{<stała>} \\ \text{<wyrażenie arytmetyozne>} \end{array} \right\}$$

Jeśli relaoja w instrukoji jest spełniona, wykonywany jest wiersz o numerze podanym w instrukoji, jeśli nie, wykonywany jest wiersz następny za instrukoją warunkową.

W systemie MERA instrukoja warunkowa ma postać:

$$\text{IF <operand> <operator relaoji> <operand>} \left\{ \begin{array}{l} \text{THEN} \\ \text{GOTO} \\ \text{GOSUB} \end{array} \right\} \text{<nr wiersza>}$$

$$\text{<operand>} ::= \left\{ \begin{array}{l} \text{<zmienna liozbowa>} \\ \text{<stała liozbowa>} \\ \text{<wyrażenie arytmetyozne>} \end{array} \right\}$$

Jest rozszerzona o możliwość wykonania skoku ze śladem (GOSUB). Postać IF - THEN jest równoważna postaoi IF - GOTO.

Instrukoja wejścia

Wprowadzanie danych w czasie działania programu realizuje instrukcja INPUT. Postać ogólna instrukcji:

INPUT [<stała alfanumeryczna>,] <zmienna1> [,{<zmienna2>} ...]

Podczas wykonywania instrukcji na końcówce wyświetlona zostanie <stała alfanumeryczna> i znak ?. Następnie program czeka na podanie wartości dla zmiennych występujących na liście zmiennych.

Przykład

INPUT "WPROWADZ DANE", A

Na ekranie wyświetli się napis:

WPROWADZ DANE ?

Gdy napiszemy 12.5 [CR], na zmienną A wczytana zostanie wartość 12.5.

W systemie MERA-400 instrukcja wejścia ma postać:

INPUT <zmienna liczbowa> [,{<zmienna liczbowa>} ...]

Nie ma w niej parametru <stała alfanumeryczna>, zatem gdy chcemy zaznaczyć, że program czeka na wprowadzenie danych, należy użyć sekwencji instrukcji:

{PRINT / PRINTUSING}
INPUT .....

Przykład

PRINT "WPROWADZ DANE"

INPUT A

Te dwie instrukcje są równoważne jednej instrukcji z przykładu poprzedniego.

Instrukcje wyjścia

Wyniki obliczeń programu wyprowadza się na urządzenie wyjściowe (monitor, drukarka) instrukcjami PRINT lub PRINTUSING.

Instrukcja PRINT realizuje drukowanie według przyjętego w systemie formatu. Gdy proponowane przez system rozmieszczanie wyników obliczeń na wydruku jest niezadowalające, można określić swój format wydruków korzystając z instrukcji PRINTUSING.

Postać ogólna instrukcji PRINT jest następująca:

PRINT <element> [{{;/,} <element>} ....] [;/,]

<element> ::= <stała> / <zmienna> / <wyrażenie arytmetyczne> / TAB(<wyrażenie>)

Użycie separatora , między elementami powoduje rozpoczęcie drukowania elementu od początku 15-znakowej strefy, na jakie dzieli się pole przeznaczone do wydruku.

Separator ; likwiduje podział wydruku na strefy. Można korzystać wtedy z funkcji TAB, która ustala początek wydruku elementu na pozycji wyznaczonej przez argument funkcji.

Wartości liczbowe drukowane są w dwóch formatach:

- dla wartości < 0.1 lub $> 10^{13}$

  $\pm$ M.MMMMMMMME$\pm$XX

- dla wartości z przedziału $\left[0.1,\ 10^{13}\right]$

  ±MMMMMM.MMMMMM

  Liczba może zajmować maksymalnie 14 znaków.

W systemie BASIC MERA-400 instrukcja PRINT działa podobnie. Jedyna różnica polega na tym, że drukowane liczby zawierają mniej cyfr (maksymalnie sześć cyfr). Postać drukowanej liczby jest następująca:

±M.MMMMME±XX lub ±MMM.MMM

Postać ogólna instrukcji PRINTUSING jest następująca:

PRINTUSING < nr wiersza > [, { < element > [; / ,] } ...]

Instrukcja drukowana używa instrukcji formatującej znajdującej się w < nr wiersza > . Instrukcja formatująca (wzorca) ma postać:

% [< łańcuch alfanumeryczny >] [{ [{ + / - / $ }] [# [,] ...] [. [# ...]] [↑↑↑↑]

< łańcuch alfanumeryczny > } ....]

Znak # oznacza pozycję cyfry.

Instrukcja wzorca może zawierać teksty umieszczone przed, między i po specyfikacjach formatu. Przykłady formatów dla liczb są następujące:

- liczby całkowite # # # (drukowanie liczb co najwyżej trzycyfrowych)
- liczby rzeczywiste # # . # # # (drukowanie liczb o dwóch cyfrach przed kropką dziesiętną i trzech po kropce)
- liczby rzeczywiste w postaci wykładniczej # . # # ## ↑↑↑↑ (strzałki oznaczają pole przeznaczone na wydruk E±XX)

Instrukcja drukowania (PRINTUSING i wzorzec) systemu MERA-400 różni się następującymi szczegółami:

- po < nr wiersza > w PRINTUSING występuje separator : ,
- instrukcja wzorca rozpoczyna się znakiem : , a nie % ,
- pole wykładnika zajmuje 5 znaków
- przy drukowaniu następuje zaokrąglanie liczby

Przykład

Niech zmienna A ma wartość 12.7886.

Drukując instrukcję PRINTUSING ... ,A, dla której we wzorcu wyspecyfikowano format # # . # # # otrzyma się wynik:

- w systemie MERA 12.789,
- w systemie WANG 12.788.

W systemie BASIC WANG należy używać instrukcji:

PRINTUSING ... , A+Ø.ØØØ5

Instrukcja komentarza

Instrukcja komentarza jest używana w celu zwiększenia czytelności programu. Zawiera dowolny tekst objaśniający działanie programu, wskazówki dla konserwatora programu itp. Pisząc komentarze, należy pamiętać, że instrukcja ta, chociaż jest ignorowana podczas wykonywania programu zajmuje miejsce w pamięci. Dla właściwego programu zostaje zatem mniej miejsca w PAO.

Postać ogólna instrukcji komentarza jest następująca:

REM < łańcuch alfanumeryczny >

Przykład

```
1Ø  REM PROGPAM KATLG
2Ø  REM ZAKLADA KATALOG ELEMENTOW WSPORCZYCH
```

Instrukoje deklaracji

Instrukoje należące do tej klasy są instrukojami niewykonywalnymi, tzn. z punktu widzenia programisty nie powodują żadnej akoji. Odgrywają one zasadniczą rolę z punktu widzenia systemu, dostarozająo mu opisu obiektów, na któryoh będą wykonywane operaoje.

Na podstawie opisu system rezerwuje odpowiednie obszary w pamięoi operaoyjnej i przy odwoła-niu do tych obszarów odpowiednio interpretuje jego zawartość.

Postać ogólna instrukoji jest następująoa:

DIM < lista zmiennych >
COM < lista zmiennych >

Instrukoja DIM służy do rezerwowania obszarów w PAO dla tablic jedno- i dwuwymiarowych, a tak-że dla zmiennych alfanumeryoznych.

Przykład

```
1ØØ   DIM N1$16, D(36,12)
```

Instrukoja powyższa spowoduje zarezerwowanie odpowiedniej liozby bajtów na zmienną alfanume-ryczną N1$ o długośoi 16 znaków i tablioę liozbową o 432 elementaoh - zarezerwuje 3872 bajtów.

Instrukoja COM umożliwia przenoszenie wartośoi danyoh między segmentami dużego programu. Każ-dy segment musi zaczynać się od takiej samej instrukoji COM, która listą zmiennyoh określa ob-szar wspólny (COMMON) dla segmentów. Zmienne zadeklarowane w COM nie będą wymazywane z PAO przy ładowaniu do niej nowego segmentu programu.

Przykład

```
1Ø  COM  T$(5)44,A1(26),A1
```

Instrukoja deklaruje tablicę alfanumeryozną T (5 elementów po 44 znaki każdy), tablioę liozbową (wektor) A1 o 26 elementach i liozbową zmienną prostą A1. Zmienne te nie będą wymazywane przy ściąganiu segmentu programu do pamięoi operacyjnej.

W systemie BASIC MERA-400 dopuszcza się używanie jedynie instrukoji DIM. Instrukoja COM jest niedozwolona.

Instrukoja zatrzymania

Postać ogólna instrukoji zatrzymania realizaoji programu jest następująoa:

STOP ["<łańouoh alfanumeryozny>"]

Instrukoja powoduje zatrzymanie programu - możliwe jest wtedy wykonanie pewnyoh ozynnośoi manualnyoh, np. zmiana kasety, włączenie drukarki lub wykonanie dyrektyw systemowyoh, np. rezer-waoja obszaru na dysku. Aby wznowić program od następnej instrukoji po STOP, używa się dyrektywy CONTINUE.

Zatrzymanie się programu sygnalizowane jest wyświetleniem słowa STOP i łańouoha alfanumeryoz-nego. W programie może występować kilka instrukoji zatrzymania, parametr instrukoji znakomioie ułatwia rozpoznanie, w którym miejsou wstrzymano działanie programu.

Przykład

```
35Ø  STOP "POTRZEBNA KASETA PB4"
```

Na ekranie zostanie wyświetlony napis: STOP POTRZEBNA KASETA PB4 i program zawiesi swoją działalność.

W systemie MERA-400 instrukcja STOP służy jedynie do zakończenia wykonywania programu. Jest instrukcją bez parametru o postaci: STOP

Przykład

```
2ØØ    STOP
```

Działanie programu jest zakończone, na urządzeniu zwrotnym WE/WY system wydrukuje informację w postaci:

```
FINISH 2ØØ
```

Instrukcja końca programu

Instrukcja końca programu zatrzymuje odpowiednią ścieżkę realizacji programu. Zatrzymanego tą instrukcją programu nie można wznowić dyrektywą CONTINUE .

Postać ogólna dyrektywy jest następująca:

```
END
```

Przykład

```
1Ø     REM PGM TP1
 .
 .
 .
8Ø     IF A=Ø THEN 1ØØ
9Ø     END
 .
 .
 .
1ØØ/   LET A=B
...
2ØØØ   END
```

Program zakończy się na wierszu o numerze 9Ø lub o numerze 2ØØØ w zależności od wyników relacji A=Ø (wiersz 8Ø).

W systemie BASIC MERA-400 instrukcja END jest logicznym końcem programu. Wiersze o numerach wyższych od wiersza z instrukcją END są przez translator pomijane.

Przykład

```
1Ø    REM PGM1
 .
 .
9Ø    END
1ØØ   REM PGM2
 .
 .
19Ø   END
```

Mając powyższy program w PAO, wywołując go od wiersza 1ØØ system zachowa się następująco:

```
RUN 1ØØ          (wywołanie programu)
RUNNING XXX      (XXX - zajętość pamięci)
FINISH 8Ø        ( zatrzymanie na wierszu 8Ø zamiast na 18Ø )
```

Instrukcja END działa tak samo, jak instrukcja STOP.

Instrukcje współpracy z pamięciami zewnętrznymi

W systemie BASIC MERA nie przewidziano możliwości korzystania z pamięci zewnętrznych w takiej postaci, jak na maszynie cyfrowej WANG 2200. Będzie więc jedynie przedstawiona instrukcja umożli-

wiająca segmentowanie programu w systemie BASIC WANG w oelu ilustraoji rozwiązania problemu wykonywania dużych programów.

Postać ogólna instrukoji jest następująca:

$$\text{LOAD}\left[\ \text{DC}\ \left\{\begin{matrix}\text{R}\\\text{F}\end{matrix}\right\}\right]\left[\ \left\{\begin{matrix}\#\ \text{n}\\/\text{XXX}\end{matrix}\right\}\right]\ \left[\ \text{"<nazwa>"}\right]\left[\ \text{<nr1>}\right]\left[\ ,\text{<nr2>}\right]$$

Znaczenie parametrów instrukoji:

DC $\left\{\begin{matrix}\text{R}\\\text{F}\end{matrix}\right\}$ - ładowanie programu z dysku (R - talerz wymienny, F - stały),

n - numer logiczny zbioru, urządzenia z którego ładowany będzie program,

/XXX - adres urządzenia, z którego ładowany będzie program,

nazwa - nazwa programu, pod którą przechowywany jest program,

nr 1, nr 2 - numery wierszy określające oząść programu rezydującego w pamięci operacyjnej, która ma być usunięta a na jej miejsce załadowany nowy segment programu.

Instrukoja LOAD automatyoznie zrealizuje sekwenoję ozynności:

- STOP - zatrzymanie programu z możliwośoią wznowienia,
- CLEAR P<nr1>,<nr2> - usunięcie tekstu programu od nr 1 do nr 2,
- CLEAR N - wyzerowanie zmiennyoh spoza obszarów COMMON,
- LOAD "<nazwa>" - załadowanie nowego programu,
- RUN <nr1> - wznowienie programu od wiersza nr 1.

Przykład

```
  1Ø    REM PRZYKLAD SEGMENTACJI PROGRAMU
  2Ø    REM JEDNO Z WIELU ROZWIAZAN
   :
 17Ø    PRINT "RODZAJ PRACY:","1-MODYFIKACJA","3-KOPIA","5-OPIS"
 175    PRINT,"2-WYMAZANIE","4-DOLACZENIE","6-KONIEC
 18Ø    INPUT "NUMER",Z
 185    ON Z GOTO 21Ø,22Ø,23Ø,24Ø,25Ø,26Ø
 2ØØ    PRINT "ZLY NUMER": GOTO 17Ø
 21Ø    LOAD DC R "MODYF" 15ØØ,2ØØØ
   :
 25Ø    LOAD DC R "INFO" 15ØØ,2ØØØ
 26Ø    PRINT"!KONIEC PROGRAMU"
   :
 3ØØ    END
 31Ø    REM PROCEDURY I FUNKCJE
   :
149Ø    REM KONIEC PROC I FUNK
15ØØ    REM OBSZAR WYMIANY SEGMENTOW PROGRAMU
2ØØØ    REM KONIEC OBSZARU WYMIANY
2Ø1Ø    GOTO 17Ø
2Ø2Ø    REM KONIEC PRZYKLADU
```

W zależnośoi od wybranego rodzaju praoy z dysku ściągany jest odpowiedni program. Po wykonaniu go przechodzi się do wiersza o numerze 17Ø. Każdy program przechowywany na dysku w tym wypadku musi mieć numeraoję z przedziału [15ØØ, 2ØØØ].

Instrukoje definioji

Instrukoje należące do tej klasy pozwalają zaprogramować (jednorazowo, a wykorzystywać dowolną liozbę razy) pewnyoh p[illegible]zalnyoh w programie operacji.

Rozróżnia się dwie instrukoje:

- instrukoja definiowania funkoji,
- instrukoja definiowania podprogramu i kluoza funkoyjnego.

Instrukoja definiowania funkoji ma postaó:

DEFFN <identyfikator>(<argument>) = <wyrażenie>

<identyfikator> ::= { <litera> / <oyfra> }

<argument> ::= < liozbowa zmienna prosta>

W systemie MERA identyfikatorem może być tylko <litera> .

Aby skorzystać z funkoji, należy odwołać się do niej z różnych miejsc programu z aktualnym argumentem, np.:

A=FN <identyfikator>(<wyrażenie>)

Instrukoja definiowania podprogramu ma postać:

DEFFN' <identyfikator> { <stała alfanumeryczna / (<zmienna>[, <zmienna> ...]) }

<identyfikator> ::= liczba całkowita ØØ - 255

Po tej instrukoji występują wiersze zawierające instrukoje realizujące operację, która ma być wielokrotnie używana w programie.

Ostatnią instrukoją podprogramu musi być instrukoja RETURN.

Podprogram może być wywołany albo przez naciśnięcie klawisza funkcyjnego o numerze odpowiadającym identyfikatorowi podprogramu, albo w programie instrukoją:

GOSUB '<identyfikator> (<parametry aktualne>)

Parametry aktualne muszą odpowiadać typem i liczbą parametrów formalnych.

W systemie MERA-400 nie ma możliwości definiowania wyżej wymienionego typu podprogramu.

Funkoje standardowe

Dla ułatwienia pisania programów w języku BASIC dostępne są często używane funkoje (np. trygonometryczne) w postaci standardowej opracowanej przez producenta. Zestaw funkcji standardowych przedstawiono w tab. 6. Podano w niej znaczenie i nazwy funkcji dla obu systemów: BASIC WANG i BASIC MERA.

Tab. 6. Typowe funkoje-standardowe w systemie BASIC

Znaczenie	WANG 2200	MERA-400
sin(x)	SIN(X)	SIN(X)
cos(x)	COS(X)	COS(X)
tg(X)	TAN(X)	TAN(X)
ctg(X)	-	COT(X)
arc sin(x)	ARC SIN(X)	-
arc cos(x)	ARC COS(X)	-
arc tg(x)	ARCTAN(X), ATN(X)	ATN(X)
generator liczb losowych	RND(X)	RND
|X|	ABS(X)	ABS(X)

Znaczenie	WANG 2200	MERA-400
entier(x)	INT(X)	INT(X)
signum(x)	SGN(X)	SGN(X)
ln x	LOG(X)	LOG(X)
e^x	EXP(X)	EXP(X)
$\sqrt{x}$	SQR(X)	SQR(X)
	# PI	@ PI
1+EPS > 1 1-EPS < 1	-	EPS
największa liczba w mc	-	INF
część ułamka x	-	FRA(X)
N parzyste Ø N nieparzyste 1	-	ODD(N)

W zasadzie nie ma znacznych różnic w zbiorach funkcji standardowych omawianych systemów.

W systemie BASIC MERA funkcje trygonometryczne sin, cos, tg, ctg wymagają argumentów w radianach. W systemie BASIC WANG jednostki argumentów dla tych funkcji ustala się instrukcją:

SELECT D - argumenty w stopniach,
SELECT R - argumenty w radianach,
SELECT G - argumenty w gradach.

Przykład

```
 1Ø  REM FUNKCJE TRYGONOMETRYCZNE
  :
 5Ø  SELECT R
 6Ø  LET B=A1/ # PI
  :
1ØØ  A=SIN B
  :
2ØØ  SELECT G
  :
22Ø  G=SIN E
```

Od numeru wiersza 5Ø do 2ØØ odwołania do funkcji trygonometrycznych (np. wiersz 1ØØ) wymagają, aby argument funkcji był przeliczony na radiany. Od wiersza 2ØØ argumenty funkcji trygonometrycznych muszą być wyrażane w gradach, aby wynik obliczeń był poprawny.

Następną zasadniczą różnicą w funkcjach standardowych omawianych systemów występuje w generatorze liczb losowych.

Funkcja RND generuje liczby losowe z przedziału [0,1]. W systemie WANG jest funkcja z argumentem RND(X). Argument X ma następujące znaczenie:

X = Ø - wydawana jest pierwsza liczba losowa z "pseudolisty",
X ≠ Ø - wydawana jest kolejna liczba losowa z "pseudolisty".

Różnica ta powoduje, że w systemie MERA otrzymuje się zawsze taki sam ciąg liczb losowych, natomiast w systemie WANG można otrzymać taki sam lub różny ciąg liczb losowych (w zależności od potrzeb) za każdym przebiegiem programu.

Przykład

```
1Ø    REM GENER LICZB LOSOWYCH
2Ø    FOR I=1 TO 5
3Ø    PRINT RNL
4Ø    NEXT I
5Ø    PRINT "KONIEC GENERACJI"
6Ø    STOP
7Ø    END
```

Powyższy program w systemie MERA wydrukuje następujący ciąg liczb losowych:

```
.189292
.454239
.212831
.916276
.813Ø44
```

Powtórne uruchomienie programu spowoduje wydrukowanie takiego samego ciągu liczb losowych. Aby osiągnąć podobne działanie programu w systemie WANG program musiałby mieć postać:

```
1Ø    REM GENER LICZB LOSOWYCH
2Ø    PRINT RND(Ø)
3Ø    FOR I=1 TO 4: PRINT RND(1): NEXT I
4Ø    PRINT "KONIEC GENERACJI"
5Ø    END
```

Wyniki powyższego programu będą różne od programu:

```
1Ø    REM GENER
2Ø    FOR I=1 TO 5: PRINT RND(1): NEXT I
3Ø    PRINT "KONIEC GENERACJI"
4Ø    END
```

Struktura programu

Program w języku BASIC składa się z wierszy zawierających instrukcje przedstawione w punkcie poprzednim. Każdy wiersz opatrzony jest numerem określającym kolejność wykonywania się wierszy.

Numery w zależności od systemu zawierają się w granicach:

WANG - 1 ÷ 9999
MERA - 1 ÷ 32767

Długość wiersza programu nie może przekraczać 192 znaków, tzn. 3 wiersze na monitorze ekranowym w systemie WANG oraz 80 znaków w systemie MERA. Nie jest dopuszczalne kontynuowanie instrukcji w następnych wierszach. W wierszu musi mieścić się cała instrukcja; gdy wyrażenie arytmetyczne jest długie, trzeba podzielić go na części mieszczące się w wierszu.

W zasadzie kolejność wierszy programu wynika z kolejności wykonywania instrukcji zaplanowanej przez programistę, ale są wyjątki, które dotyczą tzw. instrukcji niewykonywalnych, dostarczających systemowi BASIC informacji, i tak:

- instrukcja COM musi występować na początku programu, może ją poprzedzać jedynie instrukcja REM,
- instrukcja DIM musi poprzedzać pierwsze odwołanie do zmiennych zadeklarowanych w instrukcji.

Natomiast deklaracje funkcji i podprogramów mogą występować w dowolnym miejscu w programie, nawet po odwołaniu się do tej funkcji i podprogramu.

Podsumowanie

System MERA-BASIC w porównaniu z systemem BASIC na maszynie cyfrowej WANG wykazuje wiele istotnych ograniczeń. Brak w nim możliwości korzystania z pamięci zewnętrznych, segmentacji progra-

mu, a w programie - możliwości korzystania ze zmiennych alfanumeryoznych itd. Ze względu na duże potencjalne możliwości samego systemu MERA-400 wydaje się celowe rozbudowanie systemu BASIC-MERA o następujące elementy:

- segmentacja programu,
- odwołanie się do programu przez nazwę,
- śledzenie wykonywania się programu (TRACE),
- wydruk użytych zmiennych w programie,
- możliwość nadawania wartości zmiennym w trybie natychmiastowym,
- korzystanie z pamięci zewnętrznych (dysk, kaseta).

System BASIC-MERA powinien zmienić charakter: z kalkulatora elektronicznego w zestaw umożliwiający użytkownikowi pracę konwersacyjną, tak jak to gwarantuje system WANG.

Literatura

[1] System MERA-400. Dokumentacja techniczno-ruchowa. Tom III. Część III. Procesory systemowe systemu SOM-3. 11. BASIC. S-OF-4-00004-00A

[2] WANG 2200. Reference Manual, 1973

[3] WANG system 2200 BASIC Programming Manual, 1973